Theoretical Computer Science 100 (1992) 325-346
Elsevier

Foundations of a theory of synchronous systems

Miklós Bartha*

Department of Computer Science, Memorial University of Newfoundland, St. John's NF, Canada A1C 5S7

Communicated by M. Nivat
Received May 1988
Revised December 1990

Abstract

Bartha, M., Foundations of a theory of synchronous systems, Theoretical Computer Science 100 (1992) 325-346.

A semantic algebra construction is introduced to model the stepwise behavior of synchronous systems in an arbitrary pointed algebraic theory T. The theory T is extended to a feedback theory $F^{\infty}T$ in which the bottom morphism is the designated point of T. The feedback theory $F^{\infty}T$ is obtained as the inverse limit of the theories n-res T that describe the stepwise behavior of systems in T restricted to the first n clock cycles. It is shown that in $F^{\infty}T$, iteration satisfies the functorial dagger condition. Some suggestions are made about how to generalize the construction to handle infinite systems.

1. Introduction

The algebraic theory of synchronous systems presented in this paper originates from a study of structural properties of these systems. The study was motivated by a simple graph model introduced in [20], according to which a synchronous system is a finite edge-weighted directed multigraph $G=(V, E)$. In the graph G, the vertices V correspond to functional elements and the edges represent interconnections between the functional elements. One vertex, called the host, is distinguished to serve as an interface for the rest of the system. The operation of the functional elements is synchronized by a common clock, and the weight of the edges expresses a length of time measured in clock cycles. This time is the delay by which data are transferred from one particular functional element to another. Since time is measured discretely, we can think of the weight n of an edge to be a buffer consisting of n consecutive registers placed along the corresponding interconnection. The register

* On leave from Bolyai Institute, A. József University, Szeged 6720, Hungary

interpretation of the weights allows us to treat the system G as independent of the clock, assuming an abstract synchronous behavior. To be able to do so, however, it must be required that in every cycle of G there exists at least one register that stops data from rippling asynchronously in the cycle. In other words, the sum of the weights of the edges constituting a cycle must always be strictly positive.

We shall use the above model as a syntactical description of synchronous systems, taking into account the following two refinements. First, we supply the vertices of the graph G with labeled input-output ports, and relate the endpoints of the edges to the ports rather than to the vertices themselves. Naturally, the source of an edge e is always an output port, and the target of e is an input port. Secondly, we split the host vertex into a number of distinguished vertices, called input-output channels. Any edge starting from the host will be started from an input channel, and the edges arriving at the host will be directed to an output channel instead. This assumption makes the description of the interface more explicit. At the same time it allows us to get rid of the unnecessary constraint that those cycles also closed up by the host, contain an edge having positive weight.

With these two modifications, the description of synchronous systems becomes very similar to that of flowchart algorithms, i.e. to flowchart schemes. This observation suggests that synchronous systems be studied in the framework of Elgot's [14] well-known model of monadic computations. The main advantage of this algebraic approach is that it separates syntax and semantics, allowing them to be treated on different levels.

The present paper is a sequel to [4], where the syntactical issues have been dealt with on an axiomatic basis. Here we address semantics, capturing the sepwise behavior of synchronous systems. To provide some details about this semantics, let us assume that the functional elements of a system G are interpreted as functions of the form $A^m \to A^n$ for some fixed set A of data objects, where m and n are the number of input and output ports, respectively. If G has q input channels and p output channels, then the semantics of G will be defined as a Mealy automaton mapping, i.e. a series of functions $g_1, g_2, \ldots, g_n, \ldots$, where $g_i : A^{i \cdot q} \to A^p$. The function g_i specifies the output of G in the ith clock cycle as depending on the inputs having arrived at the input channels of G in the ith and all the previous clock cycles. Now, if we assume that the functional elements are themselves Mealy automata, then we have to implement the scheme operations composition, sum and feedback on automaton mappings in order to specify the semantics of G again as a Mealy automaton mapping. This is the main point of the semantic algebra construction described in Section 4.

The paper is organized as follows. In Section 2 we give a short summary of the axiomatization results obtained in [4] regarding data and synchronous flowchart schemes. In Section 3 the axiomatization is extended to the semantics of schemes by introducing some further identities and conditions. Section 4 contains the main result, which is a construction extending an arbitrary pointed algebraic theory T to a feedback theory $F^{\infty}T$. The theory $F^{\infty}T$ reflects the stepwise behavior of syn-

chronous systems in the case when the functional elements are interpreted in T. The paper ends with some conclusions and generalizations in Section 5.

2. The algebra of synchronous schemes

Synchronous schemes over a doubly ranked alphabet have been introduced under the name systolic flowchart schemes in [4]. It seems, however, that the attribute "synchronous" expresses the characteristic behavior of the corresponding systems more adequately. This is the main reason for changing the original name of these objects.

Let us fix a doubly marked alphabet

$$\Sigma = \{\Sigma(n, m) \mid (n, m) \in N \times N\},$$

where N denotes the set of all nonnegative integers. A *synchronous scheme* over Σ, SΣ-scheme for short, is a finite edge-weighted directed graph G with the following additional features.

(i) Each vertex is labeled by a symbol of Σ or by one of the symbols

$$\{\mathrm{ic}_j \mid j \in [q]\} \cup \{\mathrm{oc}_i \mid i \in [p]\},$$

where p and q are fixed nonnegative integers and $[n]$ denotes the set $\{1, 2, \ldots, n\}$ for $n \in N$. The symbol ic_j (oc_i) marks the jth input channel (respectively, the ith output channel) of G, therefore the vertex wearing this label is unique for every $j \in [q]$ and $i \in [p]$. In addition, there is a unique vertex in G labeled by the symbol $\perp$, which does not occur in Σ. This vertex is called the loop vertex, and it represents a looping computation resulting in an undefined datum traditionally denoted also by $\perp$. The pair (p, q) is characteristic of the scheme G, hence we say that G is of *sort* $p \to q$.

(ii) The label of a vertex v determines the number of input and output ports associated with v in the following sense. If the label of v is a symbol in $\Sigma(n, m)$, then v has m input ports and n output ports. If the label is ic_j (oc_i), then v has only one output port (respectively, input port). Furthermore, $\perp$ associates a single output port with the loop vertex. If v has m input ports, then there are exactly m edges arriving at v, one at each input port. Similarly, each outgoing edge from v is related to some output port of v, but there can be several edges or even zero edge starting from one particular output port.

(iii) The weight of every edge is a nonnegative integer or ∞. However, the infinite weight is assigned to an edge iff it starts from the loop vertex. This suggests that the output of the loop vertex never arrives at any of the input ports of the system (scheme). Putting it alternatively, the input ports attached to the loop vertex always receive the undefined datum. The two interpretations are equivalent, since we assume that the initial contents of the registers associated with the weights is $\perp$. It is therefore not relevant from the point of view of semantics what number we actually assign as weight to an edge starting from the loop vertex. The assignment ∞ is just a practical choice based on syntactical considerations.

(iv) In every cycle of G there is at least one edge having nonzero weight.

Now we introduce the basic algebra type S that we are going to use throughout this article. The type S of $N \times N$-sorted algebras consists of composition, sum and feedback as basic operations, and 0, 1, x, ε and 0_1 as constants. The description of these operations and constants in terms of a hypothetical S-algebra

$$M = \{M(p, q) \mid (p, q) \in N \times N\}$$

is the following.

- *Composition*: This binary operation maps $M(p, q) \times M(q, r)$ into $M(p, r)$ for each triple p, q, r in N. Composition is denoted by $\cdot$ as usual.
- *Sum*: Sum is also a binary operation that maps $M(p_1, q_1) \times M(p_2, q_2)$ into $M(p_1 + p_2, q_1 + q_2)$ for every choice of the nonnegative integers p_1, p_2, q_1, q_2. Sum is denoted by $+$.
- *Feedback*: This is a unary operation mapping $M(1 + p, 1 + q)$ into $M(p, q)$ for each pair $(p, q) \in N \times N$. Feedback is denoted by $\uparrow$.
- *Constants*: $0 \in M(0, 0)$, $1 \in M(1, 1)$, $x \in M(2, 2)$, $\varepsilon \in M(2, 1)$ and $0_1 \in M(0, 1)$.

The subtype of S not containing the feedback operation is denoted D. Putting in advance that composition is associative in M, and that the elements $\sum_{i=1}^{n} 1 \in M(n, n)$, $n \in N$ serve as units for composition from both sides, we can think of M to be a category over the set of objects N. In this sense shall we use the terminology that $f: p \to q$ is a *morphism* in M, with the simple meaning that $f \in M(p, q)$. The morphism $\sum_{i=1}^{n} 1 : n \to n$ will, somewhat abusingly, be also denoted by n. Note that the constant $0 : 0 \to 0$ is covered by this notation.

Consider the sets of isomorphism classes of all SΣ-schemes (i.e. synchronous Σ-schemes) of sort $p \to q$ for all $(p, q) \in N \times N$. These sets can be given the structure of an S-algebra Sf(Σ) in the way it is defined in [4]. Here we only review the main points of that definition, leaving it to the reader to check the details. The setting-up of the algebra Sf(Σ) is based on several older well-known, flowchart scheme algebra constructions, for which the reader is referred e.g. to [15, 11, 3]. The immediate predecessor of Sf(Σ) is the S-algebra Sch(Σ) of Σ-flowchart schemes introduced in [3]. The algebra Sf(Σ) is also closely related to x-categories introduced in [18] to study the switching behavior of circuits. In that paper, Hotz used the symbol "x" in the same sense as we do it in the algebra of synchronous Σ-schemes.

The composite of two SΣ-schemes $F: p \to q$ and $G: q \to r$ is obtained by pasting them together at the input (output) channels of F (respectively, G). The weights of the merged edges are added up and the two loop vertices are identified. The sum of $G_1: p_1 \to q_1$ and $G_2: p_2 \to q_2$ is basically their parallel composition with a relabeling of the i/o channels of G_2. The feedback of $G: 1 + p \to 1 + q$ is obtained by redirecting the control from the first output channel of G to the first input channel, inserting an additional register along all the created new interconnections.

Those schemes in which the weight of every edge is zero are called *data flowchart schemes* or DΣ-schemes, for short. By the property (iv) above, every DΣ-scheme

is cycle-free. If a $D\Sigma$-scheme $p \to q$ consists only of i/o channels and the isolated loop vertex as vertices, then it can be identified with a mapping of $[p]$ into $[q]$. In this sense, the constants 0, 1, ε and 0_1 are interpreted in $\mathrm{Sf}(\Sigma)$ as the unique mappings of their respective sorts, and x is the transposition $[2] \to [2]$, see also [4, Fig. 3].

With each symbol $\sigma \in \Sigma(n, m)$ we associate a so-called *atomic* $D\Sigma$*-scheme* $|\sigma| : p \to q$ which, apart from the isolated loop vertex, consists of a single "box" labeled by σ, being connected to the i/o channels in an obvious way, see [4, Fig. 2]. It has been proved in [4] that the collection of $S\Sigma$-schemes ($D\Sigma$-schemes) is generated by the atomic schemes using the operations and constants belonging to S (respectively, D). The sub-D-algebra of $\mathrm{Sf}(\Sigma)$ consisting of all $D\Sigma$-schemes is denoted $\mathrm{Df}(\Sigma)$.

The D-algebra of mappings, i.e. $\mathrm{Df}(\emptyset)$, is an important special case, being always a subalgebra of $\mathrm{Df}(\Sigma)$. Mappings are also called base morphisms, since they can be specified as algebraic constants in appropriate D-algebras. The following mappings will play an important role in the sequel.

- $w_p(q) : p \cdot q \to q$. For any $p, q \in N$, $w_p(q)$ takes the number $(j-1) \cdot q + i$ ($j \in [p]$, $i \in [q]$) to i. Note that $w_0(q) = 0_q$ is the unique mapping $0 \to q$.
- $\kappa(n, p) : p \cdot n \to n \cdot p$. This permutation, sometimes called a *perfect shuffle*, rearranges p blocks of length n into n blocks of length p, i.e. $\kappa(n, p)$ takes $(j-1) \cdot n + i$ ($j \in [p]$, $i \in [n]$) to $(i-1) \cdot p + j$.
- $\beta \# s$. If $\beta : r \to r$ is any permutation and s is a sequence $(n_1, \ldots, n_r)$ of nonnegative integers with $n = \sum_{i=1}^{r} n_i$, then $\beta \# s : n \to n$ is the block-by-block performance of β on s, i.e. $\beta \# s$ sends $j + \sum_{i=1}^{k} n_i$, where $j \in [n_{k+1}]$ to the number $y + j$, where y is the sum of numbers n_i such that $\beta(i) < \beta(k+1)$.

Now we turn to the axiomatization of the algebras $\mathrm{Df}(\Sigma)$ and $\mathrm{Sf}(\Sigma)$. We build up three systems of identities MG, DF and SF as follows.

(1) $\mathrm{MG} = \{\mathrm{M1}, \ldots, \mathrm{M5}\}$ is the set of *magmoid identities*, cf. [2], where

M1: $f \cdot (g \cdot h) = (f \cdot g) \cdot h$ for $f : p \to q$, $g : q \to r$, $h : r \to s$;

M2: $f + (g + h) = (f + g) + h$ for $g : p_1 \to q_1$, $g : p_2 \to q_2$, $h : p_3 \to q_3$;

M3: $p \cdot f = f \cdot q = f$ for $f : p \to q$;

M4: $f + 0 = 0 + f = f$ for $f : q \to q$;

M5: $(f_1 \cdot g_1) + (f_2 \cdot g_2) = (f_1 + f_2) \cdot (g_1 + g_2)$ for $f_i : p_i \to q_i$, $g_i : q_i \to r_i$, $i = 1, 2$.

The magmoid identities were earlier considered in [8]. Saying that the identities MG are valid in a D-algebra M is equivalent to the statement that the category associated with M is *strictly monoidal*, see [22].

(2) $\mathrm{DF} = \mathrm{MG} \cup \{\mathrm{P}, \mathrm{D1}, \mathrm{D2}, \mathrm{D3}\}$, where

P: $f_1 + f_2 = x \# (p_1, p_2) \cdot (f_2 + f_1) \cdot x \# (q_2, q_1)$ for $f_i : p_i \to q_i$, $i = 1, 2$

is the block permutation axiom introduced by Elgot and Shepherdson in [15]. This axiom postulates a *symmetry* [22] for the strict monoidal category

determined by the axioms MG;

D1: $(\varepsilon+1)\cdot\varepsilon=(1+\varepsilon)\cdot\varepsilon$;

D2: $x\cdot\varepsilon=\varepsilon$;

D3: $(1+0_1)\cdot\varepsilon=1$.

(3) $\mathrm{SF}=\mathrm{DF}\cup\{\mathrm{S2},\mathrm{S2},\ldots,\mathrm{S9}\}$, where

S1: $\uparrow(f_1+f_2)=\uparrow f_1+f_2$ for $f_1:1+p_1\to 1+q_1$, $f_2:p_2\to q_2$;

S2: $\uparrow^2((x+p)\cdot f)=\uparrow^2(f\cdot(x+q))$ for $f:2+p\to 2+q$;

S3: $\uparrow(f\cdot(1+g))=(\uparrow f)\cdot g$ for $f:1+p\to 1+q$, $g:q\to r$;

S4: $\uparrow((1+g)\cdot f)=g\cdot\uparrow f$ for $f:1+q\to 1+r$, $g:p\to q$;

S5: $\uparrow 1=0$;

S6: $\varepsilon\cdot\perp=\perp+\perp$, where $\perp=\uparrow\varepsilon$;

S7: $\uparrow(f\cdot(\varepsilon+q))=\uparrow^2((\varepsilon+p)\cdot f)$ for $f:1+p\to 2+q$;

S8: $0_1\cdot\nabla=0_1$, where $\nabla=\uparrow x$;

S9: $\uparrow(\varepsilon\cdot\nabla^n)=\perp$ for all $n\in N$, where ∇^n denotes the n-fold composite of ∇.

The notation $\perp=\uparrow\varepsilon$ above is consistent with the previous meaning(s) of the symbol $\perp$, since in $\mathrm{Sf}(\Sigma)$ the algebraic constant $\uparrow\varepsilon$ is in fact the loop vertex as a self-made scheme. The following theorem has been proved in [4].

Theorem 2.1. *The algebras* $\mathrm{Df}(\Sigma)$ *and* $\mathrm{Sf}(\Sigma)$ *are freely generated by the injection* $\sigma\mapsto|\sigma|$, $\sigma\in\Sigma$ *in the variety of all D-algebras (respectively, S-algebras) satisfying the identities* DF (*respectively*, SF).

Comparing the axiomatization of $\mathrm{S}\Sigma$-schemes with that of Σ-flowchart schemes according to [3], it turns out that the difference between the algebras $\mathrm{Sf}(\Sigma)$ and $\mathrm{Sch}(\Sigma)$ on the axiomatic level is concentrated in the single identity

X: $\uparrow x=1$,

which is valid in $\mathrm{Sch}(\Sigma)$, but fails in $\mathrm{Sf}(\Sigma)$. It was proved in [3] that $\mathrm{Sch}(\Sigma)$ is freely generated by Σ in the variety of all S-algebras satisfying the identities $\mathrm{SC}=\{\mathrm{S1},\mathrm{S2},\ldots,\mathrm{S6},\mathrm{X}\}$. For a different axiomatization of flowchart schemes, see [11].

3. Algebraic, feedback and iteration theories

The most succinct way to define an algebraic theory in the sense of Lawvere [19] is to specify it as a category over the set of objects N in which the object p is the

pth copower of the object 1. Categories of this kind are called *Lawvere theories* in the literature, or just theories, for short. Using duality, we can as well define a theory to be a category in which p is the pth power of 1. The reason why we choose the coproduct interpretation in this paper is mainly historical. In fact, the product interpretation would be more suitable for the functional semantics we are dealing with, but practically all the papers written on flowchart schemes and related semantics use the coproduct formalism. Besides, the initial theory $\mathrm{Df}(\emptyset)$ of mappings is definitely coproduct oriented, which also suggests to remain faithful to the coproduct formalism.

Using the more detailed algebraic language, we define a Lawvere theory T to be an $N \times N$-sorted algebra equipped with the following operations and constants.

- *Composition*: This operation is the same as it is in type D.
- *Tupling*: For morphisms $f_i : 1 \to q$ in T, where $p, q \in N$ and $i \in [p]$, the tuple $\langle f_1, \ldots, f_p \rangle$ is a morphism $p \to q$ in T. When $p = 0$, tupling selects a constant $0_q : 0 \to q$ for each $q \in N$. Furthermore, it is understood that $\langle f_1 \rangle = f_1$.
- *Constants*: π_p^i is a constant for every $p \in N$ and $i \in [p]$.

Composition is required to be associative, and the morphisms $\langle \pi_n^1, \ldots, \pi_n^n \rangle : n \to n$, $n \in N$ are to be units for composition from both sides in T. Moreover, T must satisfy the identities TH1 and TH2 below.

TH1: $\pi_p^i \cdot \langle f_1, \ldots, f_p \rangle = f_i$ for $f_1, \ldots, f_p : 1 \to q$, $i \in [p]$;

TH2: $\langle \pi_p^1 \cdot f, \ldots, \pi_p^p \cdot f \rangle = f$ for $f : p \to q$.

Tupling can also be treated as a binary operation in T by defining

$$\langle f_1, f_2 \rangle = \langle \pi_{p_1}^1 \cdot f_1, \ldots, \pi_{p_1}^{p_1} \cdot f_1, \pi_{p_2}^1 \cdot f_2, \ldots, \pi_{p_2}^{p_2} \cdot f_2 \rangle \quad \text{for } f_1 : p_1 \to q,\ f_2 : p_2 \to q.$$

By TH1, this definition is consistent with the original understanding of tupling. If $f = \langle f_1, f_2 \rangle : p_1 + p_2 \to q$ as above, then $first_{p_1}(f)$ and $tail_{p_2}(f)$ will denote f_1 and f_2, respectively. It is also possible to define tupling directly as an associative binary operation. In this case, however, we must impose the identities $\langle f, 0_q \rangle = \langle 0_q, f \rangle = f$ for $f : p \to q$ as further requirements in addition to the above identities, to obtain an equivalent definition.

The operation sum is derived from tupling in T as

$$f_1 + f_2 = \langle f_1 \cdot (q_1 + 0_{q_2}), f_2 \cdot (0_{q_1} + q_2) \rangle \quad \text{for } f_i : p_i \to q_i,\ i = 1, 2,$$

where $q_1 + 0_{q_2}$ and $0_{q_1} + q_2$ are just short forms of

$$\langle \pi_{q_1+q_2}^1, \ldots, \pi_{q_1+q_2}^{q_1} \rangle \quad \text{and} \quad \langle \pi_{q_1+q_2}^{q_1+1}, \ldots, \pi_{q_1+q_2}^{q_1+q_2} \rangle.$$

Defining

$$0 = 0_0, \qquad 1 = \pi_1^1, \qquad x = \langle \pi_2^2, \pi_2^1 \rangle, \qquad \varepsilon = \langle \pi_1^1, \pi_1^1 \rangle$$

and adopting 0_1, a D-algebra is readily derived from T, and it is easy to see that this D-algebra satisfies the identities DF. Conversely, if a D-algebra M satisfies at

least the identities DF, then we can define the constants π_p^i in it as $\pi_p^i = 0_{i-1} + 1 + 0_{p-i}$ and derive tupling from sum as

$$\langle f_1, f_2 \rangle = (f_1 + f_2) \cdot w_2(q) \quad \text{for } f_i : p_i \to q, \ i = 1, 2.$$

Under these conditions M becomes a theory iff it satisfies the identity

TH: $\quad w_p(p) \cdot f = (p \otimes f) \cdot w_p(q) \quad$ for $f : p \to q$,

where $p \otimes f$ denotes $\sum_{i=1}^{p} f$. The reader is referred to [4, Theorem 7.1] for a proof of the above statement. By a theory map between two Lawvere theories T and T' we mean a homomorphism between the corresponding D-algebras.

Theories are well-suited semantic domains for the syntax given by DΣ-schemes. Indeed, by Theorem 2.1, every rank preserving mapping from Σ into a theory T can be extended in a unique way to a homomorphism of Df(Σ) into T. Since a DΣ-scheme $p \to q$ is usually related to a p-tuple of some functions over q variables, the assumption of the semantic algebra being a theory is quite reasonable. Note that if the semantics of Σ-schemes is defined in an algebraic theory, then the alphabet Σ can be assumed singly ranked. Any symbol $\sigma \in \Sigma(n, m)$, $n > 1$ can be replaced by a tuple of symbols $\langle \sigma_1, \ldots \sigma_n \rangle$, where $\sigma_i : 1 \to m$. In the sequel we shall assume that Σ is a fixed singly ranked alphabet, i.e. $\Sigma = \{\Sigma_n \mid n \in N\}$.

Generalizing the above idea for SΣ-schemes and S-algebras, we expect that the semantic algebras associated with synchronous schemes be theories that satisfy the identities SF, too. The question is whether these identities, i.e. SF $\cup$ TH, or perhaps a suitable extension of them, characterize the class of the desired semantic algebras in terms of axiomatization with identities or not. To be able to answer this question, we first have to work out a sufficiently large class of semantic algebras that reflect our intuitive conception about the meaning of synchronous systems. This has been outlined in [6], and the job will be completed in this article. Once such a class is given, we can compare the variety of S-algebras generated by this class with the variety induced by the proposed set of identities. If a match is found, then the answer to the above question will be positive. Otherwise the answer will be negative. The comparison can be made by checking whether the free algebras in the two classes are the same up to isomorphism. Following this method, a positive answer was found in [6] for an appropriate extension of SF $\cup$ TH. The supplementary identity, or rather a scheme of identies, has been known since 1980 as the *commutative identity*. This identity was introduced in [16] to provide an axiomatization for iteration theories.

In [14], Elgot has introduced iterative theories as a well-suited structure to define the semantics of flowchart algorithms. An iterative theory is an algebraic theory equipped with a further partial unary operation called iteration, which takes a morphism $f : p \to p + q$ into a morphism $f^\dagger : p \to q$. The class of iteration theories has been defined in [9, 10] as the variety generated by pointed iterative theories, which are essentially iterative theories in which the operation of iteration is made totally defined. It has been proved in [16] that the identities I1–I4 below serve as a basis

of identities for the variety of iteration theories.

I1: $(0_p + f)^\dagger = f$ for $f: p \to q$.

I2: $(f + 0_r)^\dagger = f^\dagger + 0_r$ for $f: p \to p + q$.

I3: $\langle f, g\rangle^\dagger = \langle f^\dagger \cdot \langle h^\dagger, q\rangle, h^\dagger\rangle$ for $f: l \to l + m + q$ and $g: m \to l + m + q$, where $h = g \cdot \langle f^\dagger, m + q\rangle: m \to m + q$.

I4: $\langle \pi_l^l \cdot \rho \cdot f \cdot (\rho_1 + q), \ldots, \pi_l^l \cdot \rho \cdot (\rho_l + q)\rangle^\dagger = \rho \cdot (f \cdot (\rho + q))^\dagger$, where $f: m \to l + q$, $\rho: l \to m$ is a surjective mapping, and each $\rho_i: l \to l$ is also a mapping such that $\rho_i \cdot \rho = \rho$.

The identities I3 and I4 are called the *pairing identity* and the *commutative identity*, respectively. The reader is referred to [11, 23, 12, 5] for other axiomatizations of iteration theories. In the last two of these references, iteration has been replaced by feedback as basic operation. The connection between the operations of iteration and feedback is expressed by the two formulas below.

$$\begin{aligned} f^\dagger &= \uparrow^p(w_2(p) \cdot f) && \text{for } f: p \to p + q; \\ \uparrow f &= (0_1 + p) \cdot (f \cdot (1 + 0_p + q))^\dagger && \text{for } f: 1 + p \to 1 + q. \end{aligned} \tag{1}$$

The identity I4 has been replaced in [5] by the following equivalent one.

C: $w_n(p) \cdot \uparrow^l f = \uparrow^{l \cdot n}(f \star (\rho_1, \ldots, \rho_l)) \cdot w_n(q)$ for $f: l + p \to l + q$,

for all $n \in N$ under every choice of the mappings $\rho_1, \ldots, \rho_l: n \to n$, where

$$f \star (\rho_1, \ldots, \rho_l) = \alpha(l, n, p)^{-1} \cdot \left(\sum_{i=1}^{n} f\right) \cdot \alpha(l, n, q) \cdot \left(\sum_{i=1}^{l} \rho_i + n \cdot q\right)$$

and $\alpha(l, n, m) = (\kappa(2, n) \# (l, m)^n) \cdot (\kappa(l, n) + n \cdot m)$,

see also Fig. 1.

The identity C is somewhat more expressive if we consider schemes as underlying syntactical objects. I4 is easier to understand if we consider a syntactical object $p \to p + q$ to be a system of p equations over $p + q$ variables, treating the last q variables as parameters when the system is solved, i.e. when the operation of iteration is performed.

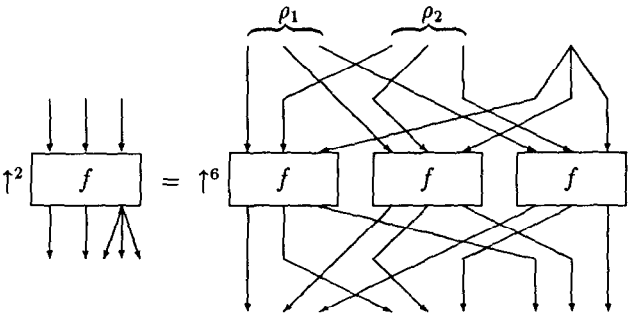

Fig. 1. The axiom C for $n = 3$, $l = 2$ and $p = q = 1$.

Turning back to the synchronous case, we adopt (1) for a definition of iteration as a derived operation in S-algebras. It must be remembered, however, that the meaning of feedback and therefore that of iteration is different now both on the syntactic and semantic level. This impels us to be cautious when trying to adopt identities containing the iteration operation. For example, we cannot adopt the pairing identity I3 in its present form, because it implies e.g. that

$$\uparrow x = (0_1 + 1) \cdot \langle \pi_3^3, \pi_3^1 \rangle^\dagger = (0_1 + 1) \cdot \varepsilon = 1.$$

We could modify I3 to meet out expectations as follows:

I3′: $\langle f, g \rangle^\dagger \cdot \langle (m \otimes \nabla) \cdot h^\dagger, q \rangle, h^\dagger \rangle$ for $f: l \to l + m + q$ and $g: m \to l + m + q$, where $h = g \cdot \langle (l \otimes \nabla) \cdot f^\dagger, m + q \rangle: m \to m + q$.

Recall that $\nabla = \downarrow x$ and $n \otimes$ stands for $\sum_{i=1}^{n}$. But we do not adopt I3′ either, because it is unnecessarily complicated for our purposes. Note that I3, too, reflects a "system of equations" point of view with respect to syntax. We do, however, adopt the identity I4. Indeed, the equivalent version C of I4 suggests that the commutative identity is valid in every reasonable semantic algebra describing the stepwise behavior of synchronous schemes, see again Fig. 1.

The axiomatizations of iteration theories appearing in [11] and [3] show that, at least in the case of flowchart algorithms, to obtain an axiomatization of the semantics it is sufficient to top the axioms describing the syntax by the two identities TH and C. This observation leaves us with the strong belief that SC $\cup$ {TH, C} is an appropriate system of identities for the semantics of synchronous schemes, even though we have not specified any class of semantic algebras yet. On an obvious analogy, the S-algebras satisfying the identities SC $\cup$ {TH, C} are called *feedback theories.*

Unfortunately, the commutative identity is rather complicated to deal with. In most of the relevant subclasses of iteration theories the following much simpler condition takes over the role of this identity.

WF: $f \cdot (\rho + q) = \rho \cdot g \Rightarrow f^\dagger = \rho \cdot g^\dagger$, where $f: l \to l + q$, $g: m \to m + q$ and $\rho: l \to m$ is a base morphism (i.e. a mapping).

The condition WF is known as the *weak functorial dagger,* and as noted in [16], I4 is a consequence of WF in every theory with iteration satisfying I1–I3. The reader is referred to [7] for a yet simpler presentation of WF in terms of the feedback operation. Feedback (iteration) theories having a weak functorial dagger are also called strong feedback (respectively, strong iteration) theories, see e.g. [7, 23]. We introduce the (full) *functorial dagger* condition in a slightly different way as opposed to its original definition in [1].

F: $f \cdot (l \otimes \nabla + q) \cdot (h + q) = h \cdot g \cdot (m \otimes \nabla + q) \Rightarrow f^\dagger = h \cdot g^\dagger$, where $f: l \to l + q$, $g: m \to m + q$ and $h: l \to m$.

It is easy to see that WF is a consequence of F. Indeed, assuming that F holds true, let $f: l \to l + q$, $g: m \to m + q$ and $\rho: l \to m$ be such that ρ is base and $f \cdot (\rho + q) = \rho \cdot g$.

Composing both sides of this equation by $(m \otimes \nabla + q)$ from the right yields

$$f \cdot (\rho + q) \cdot (m \otimes \nabla + q) = \rho \cdot g \cdot (m \otimes \nabla + q).$$

Since ρ is base, [4, Claim B*] implies that

$$(\rho + q) \cdot (m \otimes \nabla + q) = (l \otimes \nabla + q) \cdot (\rho + q).$$

Now we can apply F to obtain $f^\dagger = \rho \cdot g^\dagger$.

4. Semantics of synchronous schemes

4.1. Outline

It was observed in [4] that every SΣ-scheme $F: p \to q$ can be written in the form $F = \uparrow^l G$, where $G: l + p \to l + q$ is an appropriate DΣ-scheme. Given a theory T, the meaning of G can be specified as a morphism $g: l + p \to l + q$ in T. To extend this meaning for the scheme F we assume that T is *pointed*, i.e. it has a distinguished morphism $\perp_T : 1 \to 0$. Let us agree that by data objects we mean morphisms $1 \to 0$ in T. Then we think of F to be a Mealy automaton having as input, output and states q-tuples, p-tuples and l-tuples of data objects, respectively. The morphism $g: l + p \to l + q$ stands for the combination of the state transition and output functions, also known as the combinational logic. Assuming that the initial state is $s_0 = l \otimes \perp_T : l \to 0$, we define the meaning of F in the following natural way. In the $(i+1)$st step $(i \geq 0)$, F computes according to

$$f_{i+1} = (0_l + p) \cdot g \cdot (s_i + q)$$

as output morphism, and

$$s_{i+1} = (l + 0_p) \cdot g \cdot (s_i + q)$$

as next state morphism. Note that these morphisms determine the present output and the next state of F as "functions" of the inputs having arrived in the first i steps. By this procedure, the semantics of F as an input-output process becomes an infinite sequence of morphisms $f_1, \ldots, f_n, \ldots$, where $f_i : p \to i \cdot q$ for every $i \geq 1$. The set of such sequences will constitute the underlying set of our desired semantic algebra associated with T corresponding to sort $p \to q$. Taking $l = 0$ in the above procedure, every morphism $g: p \to q$ in T can be represented as an infinite sequence in this algebra, which in this way becomes an extension of T.

4.2. The F^∞-construction

Let T be an arbitrary pointed theory, fixed for the rest of this section. As the first step of our construction we define the theories n-res T for every $n \in N$ inductively as follows.

(i) 0-res T is the trivial pointed theory, in which 0-res $T(p, q)$ is a singleton for every $(p, q) \in N \times N$.

(ii) $(i+1)$-res $T(p, q) = i$-res $T(p, q) \times T(p, (i+1) \cdot q)$ if $i \geq 0$.

More explicitly, the morphisms $p \to q$ in n-res T are of the form

$$\langle f_1, \ldots, f_n \rangle, \tag{2}$$

where $f_i : p \to i \cdot q$ in T. Apparently, the use of the angle brackets is not legitimate in (2) because they denote tupling in T. We rectify this notation by identifying the sequence $f_1, \ldots, f_n$ with the tuple

$$\langle f_1 + 0_{(n-1)\cdot q}, f_2 + 0_{(n-2)\cdot q}, \ldots, f_n \rangle : n \cdot p \to n \cdot q \tag{3}$$

in T. Since T is a theory, the correspondence between (2) and (3) is one-to-one. The advantage of this small trick is that we can deal with morphisms in n-res T as being morphisms in T. On the other hand, the identification of (2) and (3) may sometimes lead to a confusion, in which case we write down both forms and put "$\equiv$" between them. In this setting, we define composition in n-res T to be composition in T. Tupling is, however, treated differently, hence it is necessary to use a subscript T or n-res T together with the angle brackets to mark the algebra in which this operation is to be performed. We shall nevertheless omit the subscript if it is understood from the context. We define tupling in n-res T as a binary operation in the following way.

$$\langle f, g \rangle_{n\text{-res}\,T} = \langle \langle f_1, g_1 \rangle_T, \ldots, \langle f_n, g_n \rangle_T \rangle,$$

where $f = \langle f_1, \ldots, f_n \rangle : p_1 \to q$ and $g = \langle g_1, \ldots, g_n \rangle : p_2 \to q$ in n-res T. The definition is correct, since $\langle f_i, g_i \rangle : p_1 + p_2 \to i \cdot q$ in T. The constants π_p^i and 0_q are defined by

$$(\pi_p^i)_{n\text{-res}\,T} = \langle \pi_p^i, 0_p + \pi_p^i, \ldots, 0_{(i-1)\cdot p} + \pi_p^i \rangle;$$

$$(0_q)_{n\text{-res}\,T} = \langle 0_q, 0_{2\cdot q}, \ldots, 0_{n\cdot q} \rangle.$$

Proposition 4.1. *n-res T is a theory for every $n \in N$.*

Proof. We can assume that $n \geq 1$. Composition is associative in n-res T, being the same as composition in T. Since

$$\langle \pi_p^1, \ldots, \pi_p^p \rangle_{n\text{-res}\,T} = \langle p, 0_p + p, \ldots, 0_{(n-1)\cdot p} + p \rangle \equiv (n \cdot p)_T,$$

these morphisms serve as units for composition from both sides. Tupling is clearly associative, and we also have

$$\langle f, 0_q \rangle = \langle 0_q, f \rangle = f \quad \text{for } f : p \to q$$

in n-res T. The proofs of the identities TH1 and TH2 are routine computations and they are left to the reader. □

Although we know how to derive sum from tupling in a theory, we give a separate definition of + in n-res T because it is one of our basic operations. Thus,

$$f+g=\langle f_1+g_1,\ldots,f_n+g_n\rangle\cdot\kappa(n,2)\#((q_1)^n,(q_2)^n)$$

for $f=\langle f_1,\ldots,f_n\rangle: p_1\to q_1$ and $g=\langle g_1,\ldots,g_n\rangle: p_2\to q_2$ in n-res T. The reader is invited to check the correctness of this definition.

As a category, n-res T is a subcategory of T over the objects $\{n\cdot p\,|\,p\in N\}$. In this subcategory, the object $n\cdot p$ is still the pth copower of the object n $(=n\cdot 1)$, but the coproduct injections in n-res T are different from those in T. The theory T is injected into n-res T by the endofunctor $n\otimes: T\to T$ which sends the object p to the object $n\cdot p$ and a morphism $f: p\to q$ to the morphism $n\otimes f: n\cdot p\to n\cdot q$. Viewing $n\otimes$ as a functor $T\to n$-res T between theories, it is easy to see that it preserves coproducts, i.e. it is a theory map. In terms of D-algebras, $n\otimes$ is an embedding of T into n-res T. If T' is another theory and $\psi: T\to T'$ is a theory map, then the theory map n-res $\psi: n$-res $T\to n$-res T' is just the restriction of ψ to the morphisms belonging to n-res T.

Note that it is also possible to define a theory based on the full subcategory of T over the objects $\{p\cdot n\,|\,p\in N\}$ so that $p\cdot n$ becomes the pth copower of n $(=1\cdot n)$ with the coproduct injections inherited from T. This theory is known as n-dil T, cf. [2], and the connection between n-res T and n-dil T is formulated in [6, Lemma 2].

There is a natural homomorphism $\phi_n: (n+1)$-res $T\to n$-res T for each $n\in N$ defined by

$$\phi_n(\langle f_1,\ldots,f_{n+1}\rangle)=\langle f_1,\ldots,f_n\rangle$$

for $f=\langle f_1,\ldots,f_{n+1}\rangle$ in $(n+1)$-res T. Intuitively, ϕ_n forgets the $(n+1)$st step of the computation. The homomorphisms ϕ_n introduce an inverse system over the theories n-res T, $n\in N$.

The second step of the construction is to make n-res T a feedback theory. For technical reasons we develop the operations feedback and iteration in parallel for n-res T, using in our computations whichever is more convenient. We start with feedback. For $f=\langle f_1,\ldots,f_n\rangle: l+p\to l+q$ in n-res T, define a sequence of morphisms $u_i^l(f): l+p\to i\cdot q$, $0\leq i\leq n$ in T inductively as follows.

$$u_0^l(f)=(l+p)\otimes\perp_T;$$

$$u_i^l(f)=f_i\cdot\langle first_l(u_0^l(f))+q,\ldots,first_l(u_{i-1}^l(f))+q\rangle \quad \text{if } i\in[n].$$

Recall that $f_i: l+p\to i\cdot(l+q)$ in T and

$$\left\langle \begin{matrix} i \\ j=1 \end{matrix}\right\rangle(first_l(u_{j-1}^l(f))+q)\equiv\left\langle \begin{matrix} i \\ j=1 \end{matrix}\right\rangle(first_l(u_{j-1}^l(f))+q+0_{(i-j)\cdot q}).$$

The morphism

$$up^l(f)=\langle tail_p(u_1^l(f)),\ldots,tail_p(u_n^l(f))\rangle$$

will define $\uparrow^l f$ in n-res T, but to legitimize this definition we need to prove that

$$up^1(up^l(f)) = up^{l+1}(f)$$

for $f: l+1+p \to l+1+q$ in n-res T.

Now we introduce iteration. For $g = \langle g_1, \ldots, g_n \rangle : p \to p+q$ in n-res T, define the sequence of morphisms $d_i(g): p \to i \cdot q$ in T in the following way.

$$d_0(g) = p \otimes \perp_T;$$

$$d_i(g) = g_i \cdot \langle d_0(g)+q, \ldots, d_{i-1}(g)+q \rangle \quad \text{if } i \in [n].$$

Let

$$dag(g) = \langle d_1(g), \ldots, d_n(g) \rangle.$$

We use $dag(g)$ only as a temporary notation for $g^\dagger$, until we are able to write $\uparrow^l$ for up^l. Remember that our basic algebra type is S, that is why we keep dealing with iteration as a derived operation. The following lemma states the expected connection between feedback and iteration.

Lemma 4.2. *In n-res T we have*

$$up^l(f) = (0_l + p) \cdot dag(f \cdot (l + 0_p + q)) \quad \text{for } f: l+p \to l+q;$$

$$dag(g) = up^p(w_2(p) \cdot g) \quad \text{for } g: p \to p+q.$$

Proof. A straightforward induction on i shows that for every $0 \leq i \leq n$,

$$u_i^l(f) = d_i \cdot (f \cdot (l + 0_p + q));$$

$$d_i(g) = u_i^p(g),$$

which implies the assertion of the lemma by the definitions of $up^l(f)$ and $dag(g)$. □

Lemma 4.3. *For every $f = \langle f_1, \ldots, f_n \rangle : l+1+p \to l+1+q$ in n-res T and every $0 \leq i \leq n$,*

$$u_i^{l+1}(f) = u_i^l(f) \cdot \langle \pi^{l+1} \cdot u_0^{l+1}(f) + q, \ldots, \pi^{l+1} \cdot u_{i-1}^{l+1}(f) + q \rangle.$$

Proof. By induction on i. The case $i = 0$ is trivial. If $i \geq 1$, then

$$\begin{aligned} u_i^{l+1}(f) &= f_i \cdot \langle first_{l+1}(u_0^{l+1}(f)) + q, \ldots, first_{l+1}(u_{i-1}^{l+1}(f)) + q \rangle \\ &= f_i \cdot \langle \langle first_l(u_0^{l+1}(f)) + 0_q, \ldots, first_l(u_{i-1}^{l+1}(f)) + 0_q \rangle, \\ &\qquad \langle \pi^{l+1} \cdot u_0^{l+1}(f) + q, \ldots, \pi^{l+1} \cdot u_{i-1}^{l+1}(f) + q \rangle \rangle_{i\text{-res } T}. \end{aligned}$$

Define

$$t = \langle \pi^{l+1} \cdot u_0^{l+1}(f) + q, \ldots, \pi^{l+1} \cdot u_{i-1}^{l+1}(f) + q \rangle$$

as a shorthand. By the induction hypothesis,

$$u_j^{l+1}(f) = u_j^l(f) \cdot \langle \pi^{l+1} \cdot u_0^{l+1}(f) + q, \ldots, \pi^{l+1} \cdot u_{j-1}^{l+1}(f) + q \rangle$$

for all $0 \leqslant j \leqslant i-1$. Putting it equivalently,

$$u_j^{l+1}(f) + 0_{(i-j)\cdot q} = (u_j^l(f) + 0_{(i-j)\cdot(1+q)}) \cdot t.$$

Thus, the above derivation can be continued as

$$\begin{aligned}
u_i^{l+1}(f) &= f_i \cdot \left\langle \left\langle \begin{matrix} i-1 \\ j=0 \end{matrix} \right\rangle (first_l(u_j^{l+1}(f)) + 0_q),\ t \right\rangle_{i\text{-res } T} \\
&\equiv f_i \cdot \left\langle \left\langle \begin{matrix} i-1 \\ j=0 \end{matrix} \right\rangle (first_l(u_j^{l+1}(f)) + 0_{(i-j)\cdot q}),\ t \right\rangle_{i\text{-res } T} \\
&= f_i \cdot \left\langle \left\langle \begin{matrix} i-1 \\ j=0 \end{matrix} \right\rangle (first_l((u_j^l(f) + 0_{(i-j)\cdot(1+q)}) \cdot t)),\ t \right\rangle_{i\text{-res } T} \\
&= f_i \cdot \left\langle \left(\left\langle \begin{matrix} i-1 \\ j=0 \end{matrix} \right\rangle (first_l(u_j^l(f)) + 0_{(i-j)\cdot(1+q)}) \right) \cdot t,\ t \right\rangle_{i\text{-res } T} \\
&\equiv f_i \cdot \left\langle \left(\left\langle \begin{matrix} i-1 \\ j=0 \end{matrix} \right\rangle first_l(u_j^l(f)) \right) \cdot t,\ t \right\rangle_{i\text{-res } T} \\
&= f_i \cdot \left\langle \left\langle \begin{matrix} i-1 \\ j=0 \end{matrix} \right\rangle first_l(u_j^l(f)),\ (1+q)_{i\text{-res } T} \right\rangle_{i\text{-res } T} \cdot t \\
&= f_i \cdot \left(\left\langle \begin{matrix} i-1 \\ j=0 \end{matrix} \right\rangle (first_l(u_j^l(f)) + 1 + q) \right) \cdot t \\
&= u_i^l(f) \cdot t. \qquad \square
\end{aligned}$$

Lemma 4.4. *For f and i as in Lemma* 4.3,

$$u_i^1(up^l(f)) = tail_{1+p}(u_i^{l+1}(f)).$$

Proof. Once more, by induction on i. The case $i=0$ is again trivial. If $i \geqslant 1$, then by definition

$$u_i^1(up^l(f)) = tail_{1+p}(u_i^l(f)) \cdot \left\langle \begin{matrix} i-1 \\ j=0 \end{matrix} \right\rangle (first_1(u_j^1(up^l(f))) + q).$$

From the induction hypothesis we have

$$first_1(u_j^1(up^l(f))) = \pi^{l+1} \cdot u_j^{l+1}(f)$$

for all $0 \leqslant j \leqslant i-1$. Consequently,

$$u_i^1(up^l(f)) = tail_{1+p}(u_i^l(f)) \cdot t,$$

where t is the morphism used in the proof of Lemma 4.3. The assertion now follows from Lemma 4.3. $\square$

Theorem 4.5. *The identity*

$$up^{l+1}(f) = up^1(up^l(f)) \quad \text{for } g: l+1+p \to l+1+q$$

is valid in n-res T.

Proof.

$$\begin{aligned} up^{l+1}(f) &= \left\langle \begin{matrix} n \\ i=1 \end{matrix} \right\rangle tail_p(tail_{1+p}(u_i^{l+1}(f))) \\ &= \left\langle \begin{matrix} n \\ i=1 \end{matrix} \right\rangle tail_p(u_i^1(up^l(f))) \qquad \text{(Lemma 4.4)} \\ &= up^1(up^l(f)) \qquad \text{(def.)} \quad \square \end{aligned}$$

Theorem 4.5 allows us to define feedback in n-res T by the formula

$$\uparrow^l f = up^l(f) \quad \text{for } f: l+p \to l+q.$$

Then, according to Lemma 4.2, iteration is derived in n-res T as

$$g^* = dag(g) \quad \text{for } g: p \to p+q.$$

For a morphism $p \to p+q$ in an arbitrary theory with feedback, the *Kleene sequence* $g^{(k)}$, $k \geq 0$ of g is defined in the following way.

$$g^{(0)} = p \otimes \bot + 0_q;$$

$$g^{(j+1)} = g \cdot \langle (p \otimes \nabla) \cdot g^{(j)}, q \rangle \quad \text{if } f \geq 0.$$

Note that $\bot = \uparrow \varepsilon$ and $\nabla = \uparrow x$ above, as there is no other meaningful interpretation of these symbols in the present context.

Theorem 4.6. *The identity*

$$g^* = g^{(k)} \quad \text{for } g: p \to p+q$$

is valid in n-res T *for every* $k \geq n$.

Proof. By induction on n. The basis case $n = 0$ is obvious, since 0-res T is the trivial pointed theory. Suppose that $n \geq 1$, and let $g = \langle g_1, \ldots, g_n \rangle$ be a morphism $p \to p+q$ in n-res T. In our argument we shall make use of the theory map $\phi_{n-1}: n\text{-res } T \to (n-1)\text{-res } T$ introduced earlier in this section. Recall that ϕ_{n+1} "forgets the nth step of the computation". Observe that

$$\phi_{n-1}(dag(g)) = dag(\phi_{n-1}(g))$$

holds by definition, that is, ϕ_{n-1} is a homomorphism of S-algebras as well. For this reason we also have

$$\phi_{n-1}(g^{(k-1)}) = (\phi_{n-1}(g))^{(k-1)}.$$

On the other hand,

$$(\phi_{n-1}(g))^{(k-1)} = (\phi_{n-1}(g))^{\dagger}$$

comes from the induction hypothesis. Putting these facts together we obtain

$$\phi_{n-1}(g^{(k-1)}) = \phi_{n-1}(g^{\dagger}) = \langle d_1(g), \ldots, d_{n-1}(g)\rangle.$$

Some routine computation shows that

$$\begin{aligned}(p\otimes\nabla)_{n\text{-res }T} &= \langle p\otimes\perp_T + 0_p, p+0_p, 0_p+p+0_p, \ldots, 0_{(n-2)\cdot p}+p+0_p\rangle \\ &\equiv p\otimes\perp_T + (n-1)\cdot p + 0_p.\end{aligned}$$

Thus, if $g^{(k-1)} = \langle t_1, \ldots, t_n\rangle$ with $t_i = d_i(g)$ for every $i \in [n-1]$, then

$$\begin{aligned}g^{(k)} &= g\cdot\langle (p\otimes\nabla)\cdot g^{(k-1)}, q\rangle_{n\text{-res }T} \\ &= g\cdot\langle\langle p\otimes\perp_T + 0_q, t_1+0_q, \ldots, t_{n-1}+0_q\rangle, q\rangle_{n\text{-res }T} \\ &= g\cdot\langle d_0(g)+q, \ldots, d_{n-1}(g)+q\rangle \\ &= g^{\dagger}. \qquad \square\end{aligned}$$

Proposition 4.7. *In n-res T, iteration satisfies the functorial dagger condition.*

Proof. By Theorem 4.6 it is enough to prove that if $f: l \to l+q$, $g: m \to m+q$ and $h: l \to m$ are such that

$$f\cdot((l\otimes\nabla)\cdot h + q) = h\cdot g\cdot(m\otimes\nabla + q),$$

then $f^{(k)} = h\cdot g^{(k)}$ for all $k \in N$. We prove this statement by induction on k. The case $k=0$ is trivial. Assuming that $f^{(k)} = h\cdot g^{(k)}$ holds for some $k \geqslant 0$, we have

$$\begin{aligned}f^{(k+1)} &= f\cdot\langle (l\otimes\nabla)\cdot f^{(k)}, q\rangle \\ &= f\cdot\langle (l\otimes\nabla)\cdot h\cdot g^{(k)}, q\rangle \\ &= f\cdot((l\otimes\nabla)\cdot h + q)\cdot\langle g^{(k)}, q\rangle \\ &= h\cdot g\cdot(m\otimes\nabla+q)\cdot\langle g^{(k)}, q\rangle \\ &= h\cdot g\cdot\langle (m\otimes\nabla)\cdot g^{(k)}, q\rangle \\ &= h\cdot g^{(k+1)}. \qquad \square\end{aligned}$$

Our goal is to prove that n-res T is a feedback theory. Since the identities to be checked are expressed in terms of the feedback operation, it will be convenient to construct a different Kleene sequence $f_l^{[k]}: l+p \to q$, $k \in N$ for a morphism $f: l+p \to l+q$, which reflects feedback rather than iteration. The obvious construction is the following.

$$f_l^{[0]} = (l+p)\otimes\perp + 0_q;$$

$$f_l^{[j+1]} = f\cdot\langle((l\otimes\nabla)+0_p)\cdot f_l^{[j]}, q\rangle \quad \text{if } j \geqslant 0.$$

Let us agree that we omit the subscript l from $f_l^{[k]}$ if it is understood. A straightforward induction shows that for every $k \in N$,

$$f^{[k]} = (f \cdot (l + 0_p + q))^{(k)},$$

which gives us the following corollary to Theorem 4.6.

Corollary 4.8. *The identity*

$$\uparrow^l f = (0_l + p) \cdot f^{[k]} \quad \text{for } f: l + p \to l + q$$

is valid in n-res T *for every* $k \geq 1$.

Theorem 4.9. n-res T *is a feedback theory.*

Proof. We only have to prove the identities S1, S3, S4 and S9 for $n = 1$, because the remaining identities follow from the fact that n-res T is a theory and from the functorial dagger property, which n-res T has by Proposition 4.7. As to identity $\uparrow(\varepsilon \cdot \nabla) = \perp$, observe that

$$(\perp)_{n\text{-res } T} = \langle \perp_T, \ldots, \perp_T \rangle \equiv n \otimes \perp_T.$$

On the other hand,

$$\uparrow(\varepsilon \cdot \nabla) = \nabla^{\dagger} = \langle d_1(\nabla), \ldots, d_n(\nabla) \rangle.$$

Since $\nabla_{n\text{-res } T} = \langle \perp_T + 0_1, \pi^1, \ldots, \pi^{n-1} \rangle$, it is easy to check that $d_i(\nabla) = \perp_T$ for all $i \in [n]$.

The proofs of S1, S3 and S4 follow the same pattern, each of them being a simple consequence of Corollary 4.8. To show an example we prove S3 here, leaving the other two identities to the reader. For, let $f: 1 + p \to 1 + q$ and $g: q \to r$ be morphisms in n-res T. Again it is sufficient to prove that

$$(f \cdot (1 + g))^{[k]} = f^{[k]} \cdot g$$

for all $k \in N$. As in the proof of Proposition 4.7, we follow an induction on k. The case $k = 0$ is again trivial. Assuming that the statement holds for some $k \geq 0$, we have

$$\begin{aligned}
(f \cdot (1 + g))^{[k+1]} &= f \cdot (1 + g) \cdot \langle (\nabla + 0_p) \cdot (f \cdot (1 + g))^{[k]}, q \rangle \\
&= f \cdot (1 + g) \cdot \langle (\nabla + 0_p) \cdot f^{[k]} \cdot g, q \rangle \\
&= f \cdot ((\nabla + 0_p) \cdot f^{[k]} + q) \cdot \langle g, g \rangle \\
&= f \cdot \langle (\nabla + 0_p) \cdot f^{[k]}, q \rangle \cdot g \\
&= f^{[k+1]} \cdot g. \qquad \square
\end{aligned}$$

Remark. At this point the reader might have the impression that we could have avoided the tedious proof of Theorem 4.5, because that theorem has not come up in the proof of Theorem 4.9 above. In fact it has been used implicitly in the form of the identity $g^{\dagger} = dag(g)$ for $g: p \to p + q$ in n-res T. Had we dealt with iteration

as a basic operation, this connection could have served as a definition, and therefore it need not have been proved. This is true, but in this case we should have proved the pairing identity I3′ instead. We believe that Theorem 4.5, which says something very similar to I3′, was a more economical solution regarding the amount of computation needed.

As the last step of our construction we define the S-algebra $F^\infty T$ to be the limit of the inverse system $(n\text{-res } T, \phi_n)_{n\in N}$. Since identities and quasi-identities are preserved under taking the inverse limit of algebras, $F^\infty T$ is also a feedback theory having a functorial dagger. As we noticed earlier, T can be embedded into n-res T by the theory map $n\otimes : f \mapsto n \otimes f$. Clearly,

$$n \otimes f = \phi_n((n+1)\otimes f),$$

implying that T is a subtheory of $F^\infty T$ as well. Moreover, in $F^\infty T$ we have $\perp = \perp_T$, which was an implicit goal of the construction.

Given a theory map $\psi : T \to T'$, we define $F^\infty\psi : F^\infty T \to F^\infty T'$ to be the unique homomorphism of S-algebras for which the diagram in Fig. 2 commutes. This definition makes F^∞ a functor from the variety of all pointed theories to that of all feedback theories.

$$\begin{array}{ccccccccc}
T & \xleftarrow{\phi_1} & \cdots & \xleftarrow{\phi_{n-1}} & n\text{-res } T & \xleftarrow{\phi_n} & (n+1)\text{-res } T & \xleftarrow{\phi_{n+1}} & \cdots F^\infty T \\
\downarrow{\scriptstyle \psi_1=\psi} & & & & \downarrow{\scriptstyle \psi_n = n\text{-res } \psi} & & \downarrow{\scriptstyle \psi_{n+1}} & & \downarrow{\scriptstyle \psi_\infty = F^\infty\psi} \\
T' & \xleftarrow{\phi'_1} & \cdots & \xleftarrow{\phi'_{n-1}} & n\text{-res } T' & \xleftarrow{\phi'_n} & (n+1)\text{-res } T' & \xleftarrow{\phi'_{n+1}} & \cdots F^\infty T'
\end{array}$$

Fig. 2. F^∞ is a functor.

Finally, we quote a result from [6], *which answers the question raised at the beginning of this section, namely that the* S-algebras of the form $F^\infty T$ with T being a pointed theory generate the variety of feedback theories. Let $T(\Sigma_\perp)$ denote the free pointed theory generated by the ranked alphabet Σ. Furthermore, let $F_1^\infty T$ be the sub-S-algebra of $F^\infty T$ generated by T.

Theorem 4.10. $F_1^\infty(T(\Sigma_\perp))$ *is isomorphic to the free feedback theory generated by* Σ.

5. Conclusion

We have introduced a semantic algebra construction to model the stepwise behavior of synchronous systems. For an algebraic theory T and an arbitrary morphism $\perp_T : 1 \to 0$ in T, we have extended T to a feedback theory $F^\infty T$ in which

$\perp = \perp_T$. Particularly interesting cases are those in which $T = \mathrm{Pol}\, A$ is the clone algebra of some pointed Σ-algebra A. In these cases the interpretation of Σ in T is fixed by the algebra A, and $\perp_T$ is the point of A as a morphism $1 \to 0$ in Pol A. The unique extension of this interpretation to a homomorphism $\mathrm{Sf}(\Sigma) \to F_1^\infty(\mathrm{Pol}\, A)$ provides an algebraic semantics for synchronous Σ-schemes. The feedback theory $F_1^\infty(\mathrm{Pol}\, A)$ is called the *feedback algebra* induced by A. Let $T_{\Sigma_\perp}$ denote the free pointed Σ-algebra. It is known that if $\Sigma \neq \emptyset$, then $\mathrm{Pol}(T_{\Sigma_\perp}) \cong T(\Sigma_\perp)$. In view of Theorem 4.10 this implies that already the class of feedback algebras induced by pointed Σ-algebras generate the variety of feedback theories. For more details about feedback algebras, see [6].

Let $\mathscr{T}$ and $\mathscr{F}$ denote the category of all pointed theories with point preserving theory maps and the category of all feedback theories with S-algebra homomorphisms, respectively. Furthermore, let $U: \mathscr{F} \to \mathscr{T}$ be the functor which forgets the feedback operation except that $\uparrow\varepsilon$ is the designated point. Theorem 4.10 raises the following question. Is the functor F_1^∞ the left adjoint of U? If not, then what is the answer if we restrict $\mathscr{F}$ to the class of feedback theories having a functorial dagger? While the first question can be answered negatively, we do not know the answer to the second. In [7], the free strong feedback theory generated by an arbitrary theory T has been constructed as a result of a minimization procedure of expressions of the form $\uparrow^l f$, where $f: l+p \to l+q$ is a morphism in T. These expressions are called structural T-automata, cf. [17], and the resulting free algebra is denoted $F_s T$. It can be seen that in general $F_1^\infty T_\perp \not\cong F_s T$, where $T_\perp$ is the free pointed extension of the theory T. This means that even if we restrict the category $\mathscr{F}$ to feedback theories having a *weak* functorial dagger, F_1^∞ will still not be the left adjoint of the functor U.

Whereas the F^∞-construction is general enough to capture the stepwise behavior of finite synchronous schemes under all reasonable interpretations, it is not capable to handle schemes with an infinite number of boxes and/or input-output channels. For a description of infinite synchronous schemes, see e.g. [13]. Interestingly, the problem seems not to be on the semantic side. Indeed, instead of a Lawvere theory T one may as well consider an algebraic theory in the more general sense of Manes to be the subject of the F^∞-construction. Recall from [21] that an algebraic theory in the general sense is the *Kleisli category* $\boldsymbol{T}$ of a triple (T, η, μ) in the category of all sets and mappings. A Lawvere theory is the restriction of a *finitary* algebraic theory $\boldsymbol{T}$ to the objects $[n]$, $n \in N$. Observe that the algebraic theories n-res $\boldsymbol{T}$, and consequently $F^\infty \boldsymbol{T}$, are meaningful in the general case, too. One has to define the underlying functor n-res T as

$$(n\text{-res } T)A = TA \times T(2 \cdot A) \times \cdots \times T(n \cdot A)$$

for a set A and

$$(n\text{-res } T)\alpha = T\alpha \times T(2 \cdot \alpha) \times \cdots \times T(n \cdot \alpha)$$

for a mapping $\alpha: A \to B$, where $i \cdot A$ and $i \cdot \alpha$ stand for the ith copower of A and

that of α, respectively. The rest of the construction is clear. Obviously, instead of $\uparrow$ one has to consider the operations $\uparrow^A$, where A is any set. Then, for example, Theorem 4.5 can be stated as

$$\uparrow^A(\uparrow^B(f)) = \uparrow^{A+B}(f) \quad \text{for } f: A+B+C \to A+B+D$$

in n-res $\boldsymbol{T}$, where + denotes coproduct of sets. As in the finitary case, one can prove that $F^\infty \boldsymbol{T}$ satisfies the functorial dagger condition. The identities S1, S3 and S4 are also easy to formulate and prove in the general case. For example, S3 can be put as

$$\uparrow^A(f \cdot (A+g)) = (\uparrow^A f) \cdot g \quad \text{for } f: A+B \to A+C \text{ and } g: C \to D,$$

where A denotes also the morphism $\eta_{n\text{-res } T}A: A \to A$ in n-res $\boldsymbol{T}$, i.e. the mapping $\eta_T(n \cdot A): n \cdot A \to T(n \cdot A)$. The form of the identity $\uparrow(\varepsilon \cdot \nabla)$ could be

$$\nabla_A^\dagger = A^\dagger,$$

where $\nabla_A = \uparrow^A x_A$ and $x_A: 2 \cdot A \to 2 \cdot A$ is the obvious transposition morphism. We do not elaborate this point any further here.

On the level of syntax, however, one has to introduce sum as an infinitary operation, which makes the description of schemes less elegant and bit too general. A possible way to impose some regularity on infinite schemes is to generate them by appropriate grammars.

As a final point, note that the F^∞-construction allows us to unfold synchronous schemes not only in the usual vertical direction, but also horizontally. Indeed, according to [6], the semantics of a finite SΣ-scheme under the Herbrand interpretation $T_{\Sigma_\perp}$ is an infinite tuple of finite $\Sigma_\perp$-trees, i.e. an infinite D$\Sigma_\perp$-scheme. This observation, among many other reasons, makes infinite synchronous schemes worthwhile to study.

References

[1] M.A. Arbib and E.G. Manes, Partially additive categories and flow-diagram semantics, *J. Algebra* **62** (1980) 293-227.

[2] A. Arnold and M. Dauchet, Théorie des magmoïdes, *RAIRO Inform. Théor. Appl.* **12** (1978) 235-257 and **13** (1979) 135-154.

[3] M. Bartha, A finite axiomatization of flowchart schemes, *Acta Cybernet.* **2** (1987) 293-217.

[4] M. Bartha, An equational axiomatization of systolic systems, *Theoret. Comput. Sci.* **55** (1987) 265-289.

[5] M. Bartha, Connections between feedback and iteration theories, submitted for publication, 1988.

[6] M. Bartha, Interpretations of synchronous flowchart schemes, in: J. Czisik, J. Demetrovics and F. Gécseg, eds., *Proc. of the Conf. on the Fundamentals of Computation Theory, Szeged, 1989*, Lecture Notes in Computer Science Vol. 380 (Springer, Berlin, 1989) 25-34.

[7] M. Bartha, An algebraic model of synchronous systems, *Inform. and Comput.*, to appear.

[8] D.B. Benson, The basic algebraic structures in categories of derivations, *Inform. and Control* **28** (1975) 1-29.

[9] S.L. Bloom, C.C. Elgot and J.B. Wright, Solutions of the iteration equations and extensions of the scalar iteration operation, *SIAM J. Comput.* **9** (1980) 25-45.

[10] S.L. Bloom, C.C. Elgot and J.B. Wright, Vector iteration in pointed iterative theories, *SIAM J. Comput.* **9** (1980) 525-540.

[11] S.L. Bloom and Z. Ésik, Axiomatizing schemes and their behaviors, *J. Comput. System Sci.* **31** (1985) 375-292.
[12] V.E. Căzănescu and Gh. Ştefănescu, Feedback, iteration and repetition, Research Report 42, National Institute for Scientific and Technical Creation, Bucharest, 1988.
[13] K. Culik II and I. Fris, Topological transformations as a tool in the design of systolic networks, *Theoret. Comput. Sci.* **37** (1985) 183-211.
[14] C.C. Elgot, Monadic computations and iterative algebraic theories, in: H.E. Rose and J.C. Shepherdson, eds., *Logic Colloquium '73, Studies in Logic and the Foundations of Mathematics* (North-Holland, Amsterdam, 1975) 175-230.
[15] C.C. Elgot and J.C. Shepherdson, An equational axiomatization of the algebra of reducible flowchart schemes, IBM Research Report RC 8221.
[16] Z. Ésik, Identities in iterative and rational theories, *Comput. Linguistics Comput. Languages* **14** (1980) 183-207.
[17] F. Gécseg and I. Peák, *Algebraic Theory of Automata* (Akadémiai Kiadó, Budapest, 1972).
[18] G. Hotz, Eindeutigkeit und Mehrdeutigkeit formaler Sprachen, *Elektronische Informationsverarbeitung und Kybernetik* **2** (1966) 235-247.
[19] F.W. Lawvere, Functorial semantics of algebraic theories, *Proc. Nat. Acad. Sci. U.S.A.* **50**(5) (1963) 869-872.
[20] C.E. Leiserson and J.B. Saxe, Optimizing synchronous systems, *J. VLSI Comput. Systems* **1** (1983) 41-67.
[21] E.G. Manes, *Algebraic Theories* (Springer. Berlin, 1976).
[22] S. MacLane, *Categories for the Working Mathematician* (Springer, Berlin, 1971).
[23] Gh. Ştefănescu, On flowchart theories: Part I. The deterministic case, *J. Comput. System Sci.* **35** (1987) 163-191.